درس طراحی زبان‌های برنامه‌سازی

دکتر محمد ایزدی

تمرین پنجم

دانشکده مهندسی کامپیوتر

دانشگاه صنعتی شریف

نیم سال دوم ۱۴۰۰ـ۱۳۹۹

مهلت ارسال:
۸ تیر ۱۴۰۰
ساعت ۲۳:۵۹

به موارد زیر توجه کنید:

* برنامه‌های خود را به زبان Racket بنویسید.
* مهلت ارسال تمرین ساعت ۲۳:۵۹ روز ۸ تیر ۱۴۰۰ است.
* جواب خود را در غالب pdf بنویسید. در صورت نیاز می‌توانید کدهای خود را در کنار pdf بگذارید.
* در نهایت تمام فایل‌های خود را در یک فایل زیپ قرار داده و با نام $HW5\_StudentID$ در سامانه کوئرا آپلود کنید.
* هرگونه سوالی راجع به این تمرین را در زیر پست مربوطه در کوئرای درس مطرح کنید.
* در مجموعِ تمامی تمارین ۷ روز مهلت تاخیر مجاز دارید و پس از تمام شدن این تاخیرهای مجاز به ازای هر روز ۱۰ درصد از کل نمره تمرین شما کم می‌شود.
* لطفا تمرین‌ها را از یکدیگر کپی نکنید. در صورت وقوع چنین مواردی مطابق با سیاست درس رفتار می‌شود.
* بخش سوال‌های پیشنهادی نیازی به تحویل ندارد و صرفا جهت تمرین بیشتر گذاشته شده است.

نکته مهم : در سوال‌هایی که از شما خواسته می‌شود امکان جدیدی را به یک زبان اضافه کنید، باید تمام تغییرات روی گرامر، توصیف مقادیر [۱] ، محیط [۲] ، قوانین استنتاج [۳] دستورات و کدهای تغییر یافته یا اضافه شده را مشخص نمایید. (بعضی از این موارد ممکن است تغییر نکنند.) در واقع باید همه مراحل پیاده‌سازی را بنویسید. دقت کنید تغییرات روی سینتکس نیازی به بررسی ندارند.

سوال ۱

می خواهیم به زبان IMPLICIT-REFS و با حفظ نحوه پیاده‌سازی متغیرها در این زبان، امکان کار کردن مستقیم با رفرنس را نیز در اختیار برنامه نویس قرار دهیم. یعنی این زبان دستورات newref، deref و setref که در زبان EXPLICIT-REFS معرفی شدند را نیز داشته باشد. تغییرات لازم در تعاریف و قوانین را به همراه پیاده سازی توابع مفسر بیان کنید.

سوال ۲

برای هر یک از موارد زیر در زبان EXPLICIT-REFS یک توصیف [۴] بنویسید.

- عبارت begin-end که گرامر آن به شکل زیر است.

$$Expression ::= begin\ Expression\ \{;\ Expression\}^*\ end$$

این مجموعه عبارات به ترتیب اجرا می‌شوند و مقدار نهایی آخرین عبارت به عنوان جواب نهایی begin-end بازگردانده می‌شود.

- توابع new-list و set-list و get-list برای نوع داده‌ی لیست. این نوع داده لیستی است از آدرس‌ها. در واقع داده‌های ما در حافظه و لیست ما در محیط [۵] خواهد بود. در این نوع داده بر خلاف تمرین قبل،

۱ـ new-list ورودی گرفته و لیست ما طول مشخصی دارد. با ارزیابی مقدار این

[1] Specification of Values
[2] Environment
[3] Rules of inference
[4] Specification
[5] Environment

ورودی باید به یک عدد برسید (در غیر این صورت خطا رخ خواهد داد.) که طول لیست ما را مشخص می‌کند.

$$Expression \ ::= \ new-list\ (Expression)$$

۲ـ set-list و get-list تنها با اعضای درون لیست کار می‌کنند و خود لیست و طولش هیچگاه تغییر نمی‌کند.

$$Expression \ ::= \ get-list\ (Expression, Expression)$$

$$Expression \ ::= \ set-list\ (Expression, Expression, Expression)$$

get-list لیست مورد نظر و ایندکس را ورودی می‌گیرد. set-list لیست مورد نظر، ایندکس و مقدار مورد نظر را ورودی می‌گیرد. ترتیب این ورودی‌ها بر عهده خودتان است.

سوال ۳

موارد سوال قبل یعنی begin-end و لیست را پیاده‌سازی کنید. نحوه‌ی پیاده‌سازی و جزییات بیشتر بر عهده خودتان است.

سوال ۴

تغییرات زیر را در زبان IMPLICIT-REFS به وجود بیاورید:

- توابع بتوانند چند ورودی بگیرند و فراخوانی هم با چند ورودی قابل انجام باشد.
- letrec بتواند چند ورودی بگیرد و تمام توابع تعریف شده در این عبارت بتوانند از هم استفاده کنند.

دقت کنید این سوال را به گونه‌ای پیاده‌سازی کنید که تمامی فراخوانی‌ها و استفاده از متغیرها به صورت call-by-value باشد.

سوال ۵

زبان MUTABLE-PAIRS را به نحوی گسترش دهید که بتواند هر سه نوع فراخوانی call-by-value و call-by-reference و call-by-need را پشتیبانی کند. موارد زیر را در نظر داشته باشید:

- همیشه فراخوانی تابع call-by-need است. یعنی درون تابع تا به یکی از ورودی‌ها دسترسی پیدا نکنیم، مقدار آن محاسبه نمی‌شود.
- زبان باید بتواند هر دو حالت call-by-value و call-by-reference را پشتیبانی کند. یعنی در تعریف تابع proc تغییری ایجاد کنید که بتوانیم مشخص کنیم این تابع call-by-value یا call-by-reference است.

سوال ۶

می خواهیم زبان CHECKED را به گونه ای توسعه دهیم که مانند زبان C هر مقدار از نوع عدد صحیح نقش مقدار از نوع بولین نیز داشته باشد به طوری که هر جایی از یک عبارت، مقداری از نوع بولین مورد نیاز است مقدار عدد صحیح نیز مقبول است: صفر نقش false و هر عدد غیر صفر نقش true دارد. مقادیر بولین هم همچنان وجود دارند و قابل استفاده هستند. تغییرات لازم در تعریف نوع ها، ساختارهای داده ای انواع و پیاده سازی تابع type-of را بیان کنید.

سوال ۷

برای هر یک از قطعه کدهای زیر به روش استنتاج نوع، نوع مقدار نهایی عبارت و نوع هر زیرعبارت و نوع همه متغیرها را تعیین کنید یا نشان دهید عبارت از نظر نوع درست نیست:

1.
```
letrec ? even (x : ?)
    = if zero?(x) then 1 else (odd -(x, 1))
  ? odd (x : ?)
    = if zero?(x) then 0 else (even -(x, 1))
in (odd 13)
```

2.
```
let p = zero?(1) in if p then 88 else 99
```

3.
```
let f = proc (z) z in proc (x) -((f x), 1)
```

سوال ۸

می‌خواهیم زبان INFERRED را به گونه‌ای توسعه دهیم که مانند زبان C هر مقدار از نوع عدد صحیح نقش مقدار از نوع بولین نیز داشته باشد به طوری که هر جایی از یک عبارت، مقداری از نوع بولین مورد نیاز است مقدار عدد صحیح نیز مقبول است: صفر نقش false و هر عدد غیر صفر نقش true دارد. مقادیر بولین هم همچنان وجود دارند و قابل استفاده هستند. تغییرات لازم در تعریف نوع ها، ساختارهای داده ای انواع و همه پیاده‌سازی‌های لازم برای روش استنتاج نوع تا رسیدن به تابع type-of در این روش را بیان کنید.

سوال‌های پیشنهادی

سوال‌های زیر از کتاب *essentials of programming languages* توصیه می‌شود:
4.4, 4.9, 4.13, 4.19, 4.20, 4.28, 4.33, 4.42 , 7.5 , 7.8 , 7.9 , 7.10 , 7.23 , 7.24 , 7.25